МИНЕСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧЕРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«Брестский государственный технический университет»

Кафедра «Интеллектуальные информационные технологии»

Лабораторная работа №4

По дисциплине «Вычислительная математика»

За 2 семестр

Тема: «Численное интегрирование»

Выполнила:

студентка 1 курса

группы АС-56

Карпенко М.В.

Проверил:

Пролиско Е.Е.

Брест 2020

Вариант 29

Задана функция y = -3.8\*x-4.3\*cos^2(0.9\*x+2)+2.6 на интервале [-6.9, -0.9].

1. Получить оценку интеграла от заданной функции методом трапеций и методом Симпсона. Количество частичных интервалов n=6

2. Для интерполяционной квадратурной формулы использовать 7 равноотстоящих узлов.

*Код программы (методы трапеций и Симпсона):*

#include <iostream>//подключение потоковой библиотеки

#include <math.h>//подключение математической библиотеки

using namespace std;//исползование пространства имён

double f(double x) //функция, возвращающая

{ //значение заданного выражения

return -3.8 \* x - 4.3 \* pow(cos(0.9 \* x + 2), 2) + 2.6; //в данной точке

};

double simpsonIntegral(double a, double b)//функция для вычисления численного значения интеграла на заданном промежутке методом Симпсона(порабол)

{

const double width = (b - a) / 6;//ищем длинну шага

double simpson\_integral = 0;//иницилизируем переменную, для последующего вычисления значения интеграла

for (int step = 0; step < 6; step++) //цикл для вычисления интеграла, кол-во иттераций = кол-ву промежутков

{

const double x1 = a + step \* width;//значение х1 для данной точки

const double x2 = a + (step + 1) \* width;//значение х2 для данной точки

simpson\_integral += (x2 - x1) / 6.0 \* (f(x1) + 4.0 \* f(0.5 \* (x1 + x2)) + f(x2));//вычисляем интеграл от порабол на 3х узлах по формуле Симпсона

}

return simpson\_integral;//возвращаем значение функции

};

double trapezoidalIntegral(double a, double b)//функция для вычисления численного значения интеграла на заданном промежутке методом трапеций

{

const double width = (b - a) / 6;//ищем длинну шага

double trapezoidal\_integral = 0;//иницилизируем переменную, для последующего вычисления значения интеграла

for (int step = 0; step < 6; step++) //цикл для вычисления интеграла, кол-во иттераций = кол-ву промежутков

{

const double x1 = a + step \* width;//значение х1 для данной точки

const double x2 = a + (step + 1) \* width;//значение х2 для данной точки

trapezoidal\_integral += 0.5 \* (x2 - x1) \* (f(x1) + f(x2));//вычисляем площадь трапеции на данном интервале

}

return trapezoidal\_integral;//возвращаем значение функции

};

int main()

{

setlocale(0, "");

cout << "Метод Симпсона: " << simpsonIntegral(-6.9, -0.9) << endl;//вызов функции для вычисления интеграла методом Симпсона

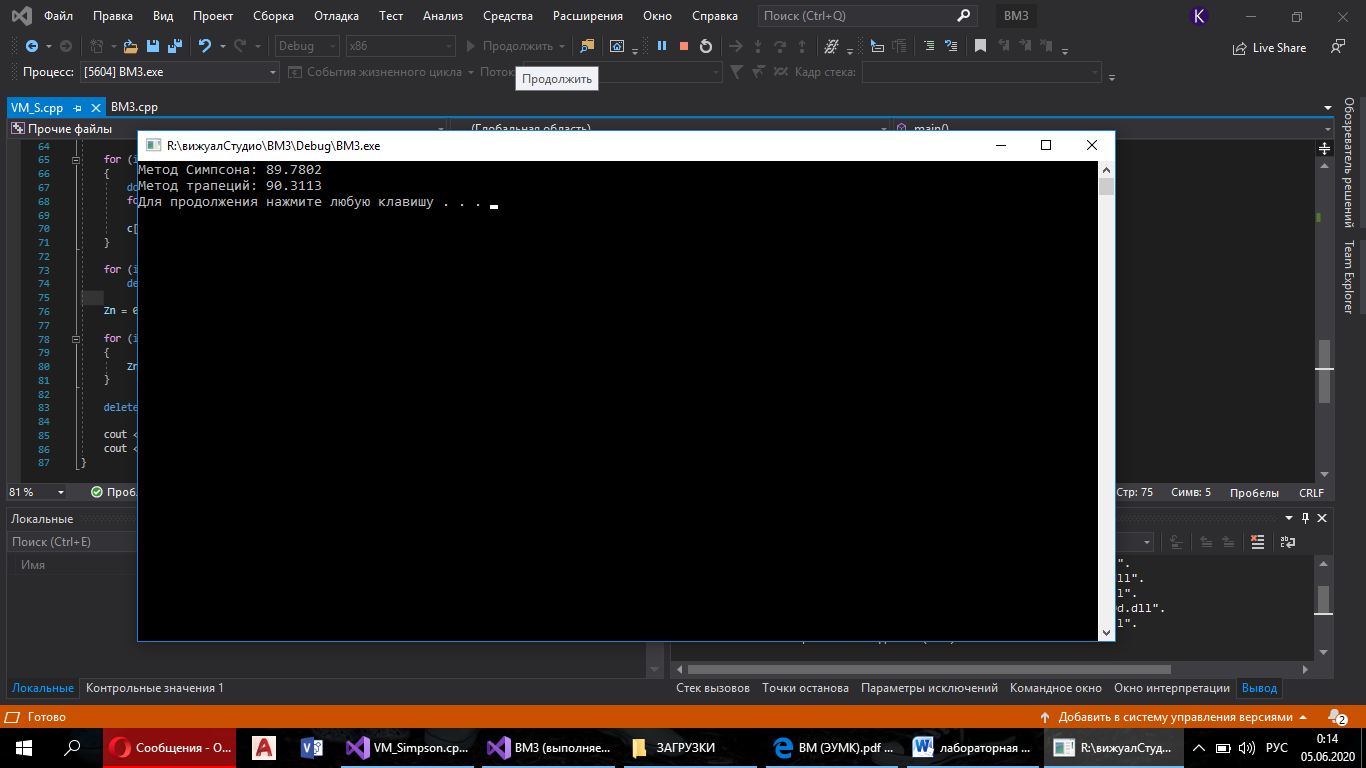
cout << "Метод трапеций: " << trapezoidalIntegral(-6.9, -0.9) << endl;//вызов функции для вычисления интеграла методом трапеций

system("pause");//пауза в программе

return 0;//возвращаемое значение - 0

}

*Результат работы программы:*



*Блок-схема программы:*



*Код программы (интерполяционная квадратурная формула):*

#include<iostream>//библиотека для организации потоков ввода-вывода

#include<windows.h>

#include <math.h> //математическая библиотека

using namespace std; //использование пространства имен std

double ff(double x) //функция, возвращающая значение заданного уравнения в точке х

{

return -3.8 \* x - 4.3 \* pow(cos(0.9 \* x + 2), 2) + 2.6; //заданное уравнение

}

int main()

{

setlocale(LC\_ALL, "rus"); //подключение русского языка

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n = 7; //количество узлов

double a = -6.9, b = -0.9; //инициализация левой и правой границ интервала соответственно

double ZnN = 89.7875814009853; //реальное значение интеграла

double Zn, h;//инициализация вещественных переменных

double\* x = new double[n]; //инициализация динамического массива

double\* y = new double[n]; //инициализация динамического массива

h = (b - a) / n; //шаг разбиения

for (int i = 0; i < n; i++) //заполнение массивов

{

x[i] = a + h \* i; //

y[i] = ff(x[i]);

}

//Метод Гаусса

int m = n + 1;

double\*\* matr = new double\* [n];

for (int count = 0; count < n; count++)

matr[count] = new double[m];

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

matr[i][j] = pow(x[j], i);

matr[i][m - 1] = (pow(b, i + 1) - pow(a, i + 1)) / (i + 1);

}

delete[]x;

for (int k = 0; k < n; k++)

{

if (matr[k][k] != 1)

for (int i = m - 1; i >= 0; i--)

matr[k][i] = matr[k][i] / matr[k][k];

if (k != n - 1)

for (int l = k + 1; l < n; l++)

for (int b = m - 1; b >= k; b--)

matr[l][b] = matr[l][b] - matr[k][b] \* matr[l][k];

}

double\* c = new double[n];

c[n - 1] = matr[n - 1][m - 1] / matr[n - 1][m - 2];

for (int i = n - 2; i >= 0; i--)

{

double sum = 0;

for (int j = m - 2; j > i; j--)

sum = sum + matr[i][j] \* c[j];

c[i] = matr[i][m - 1] - sum;

}

for (int i = 0; i < n; i++)

delete[]matr[i];

Zn = 0;

for (int i = 0; i < n; i++)

{

Zn = Zn + y[i] \* c[i]; //квадратурная формула

}

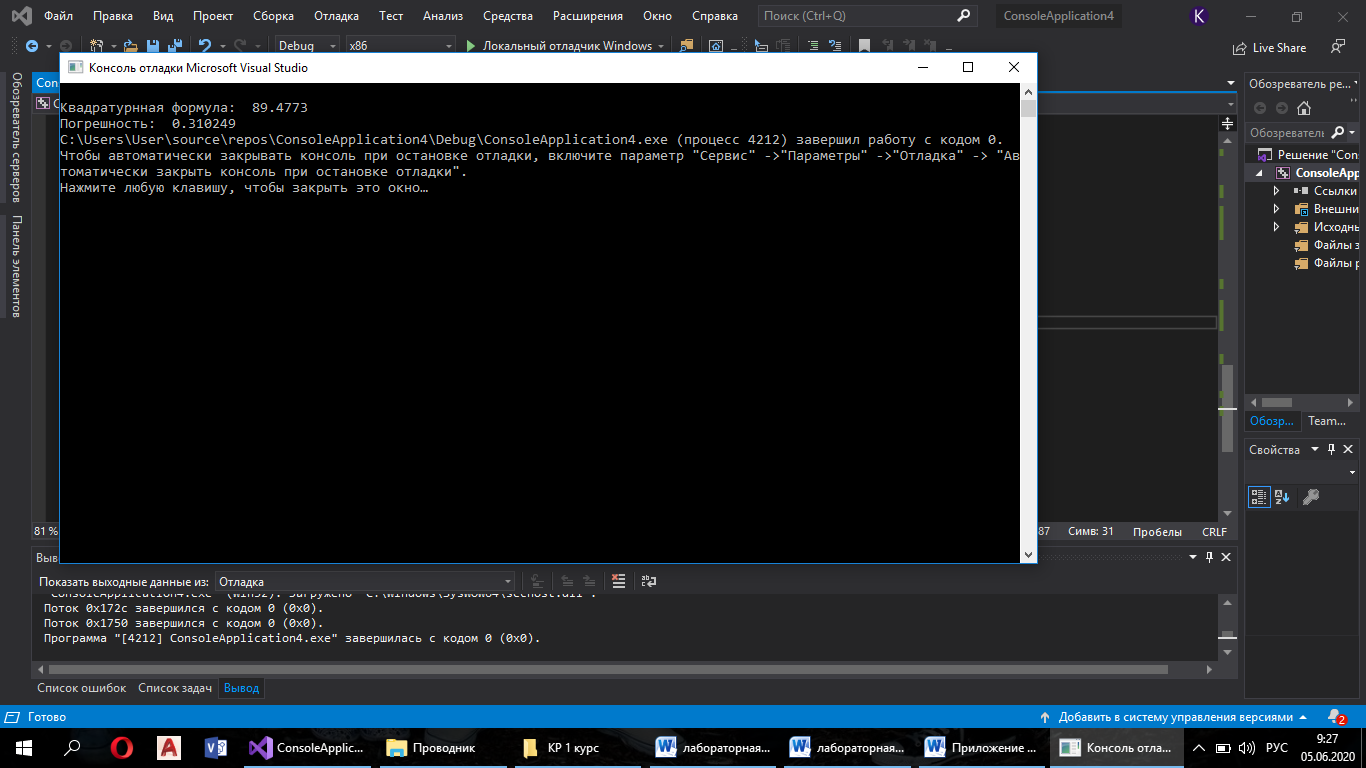
delete[]c; delete[]y;

cout << endl << "Интерполяционная формула: " << Zn << endl;

cout << "Погрешность: " << abs(Zn - ZnN);

}

*Результат работы программы:*



*Блок-схема программы:*



*Вывод:*  Получила оценку интеграла от заданной функции методом трапеций и методом Симпсона. Метод Симпсона является более точным, поскольку отклонение параболы от кривой меньше, чем хорды. Таким образом, значение, вычисленное по методу Симпсона ближе к точному значению интеграла, указанному в коде программы для квадратурной формулы.